SYSTEM AND METHOD FOR WRITING DATA FROM A STORAGE MEANS TO A MEMORY MODULE IN A SOLID STATE DISK SYSTEM

by

Richard Holzmann of Houston, Texas

BACKGROUND OF THE INVENTION

1. Field of the Invention

The invention is directed generally to solid state disk systems. More specifically, the invention is directed to a system and method for writing data from a non-volatile storage means to a volatile memory module within a solid state disk system, upon start-up.

2. Description of Related Art

The use of solid state disk (SSD) systems allows organizations to obtain increased returns from their IT hardware investments. SSD systems allow centralized storage and retrieval of data and have many advantages over individual workstations or servers that use conventional storage systems, such as conventional rotating disks or tape drives. SSDs can move much larger amounts of data and process far more I/O requests, per time period, than conventional disk systems found on most server computers. This allows users to complete single data transactions much more quickly than with conventional disk systems.

SSD systems also provide greater reliability compared to disks, reducing downtime, and they allow for centralized management of data storage and performance, thereby reducing staffing needs. The scalability of a SSD system also allows a user to build a storage area network (SAN) based upon the SSD's performance capacities. This allows for consolidated management of data storage and can create a virtual, dynamic resource that can be used for

specific tasks by separate business units, as needed. As a result, many businesses and other organizations and enterprises are incorporating SSD systems into their IT configurations.

Solid state disk systems typically comprise a temporary memory module, such as a random access memory (RAM); a battery supported power system; and a non-volatile (conventional disk) storage means. In the event of a power outage or other shutdown, data is automatically copied from the memory module to the storage means. When power is restored, the data is re-written from the storage means to the memory module upon start-up. Solid state disk systems may also comprise control devices that allow users to periodically backup data from the memory module to the storage means. Solid state disk systems may also comprise communication controllers, such as Fibre Channel (FC) controllers, Ethernet mechanisms or SCSI controllers for managing data communication with external computing devices.

Despite their many advantages, one limitation of SSD systems is the time required for data to be re-written to the memory module upon start-up. When data is copied to the storage means, it is arranged sequentially in segments, each of which consists of a range of address units. From this sequential ordering of segments, a sequential load map may be produced. Upon start-up, the storage means re-writes data segments to the memory module in the same sequential order. If a user needs to access segments 1,000,000 to 1,000,001, for example, the user must wait anywhere from five to thirty minutes before the storage means writes all of the data to the memory module. As a result, the effect of a power outage or other shutdown on the availability of the SSD system often significantly exceeds the length of the outage itself.

As a result, there is a great need in the art for a system and method for writing data from a storage means to a memory module in a solid state disk system, which allows users to immediately access needed data. The system and method must provide for accepting requests

from external devices and for writing data from the storage means to the memory module in a non-sequential order according to such requests. Once a requested segment is loaded, sequential re-writing of data from the storage means to the memory module must continue without interruption and without duplication of the requested segments.

SUMMARY OF THE INVENTION

The present invention is directed to a system and method for writing data from a storage means to a memory module in a solid state disk system, upon start-up. The invention overcomes limitations of prior SSD systems, by accepting requests from external devices and writing data from the storage means to the memory module in a non-sequential order according to such requests. According to the invention, once a requested segment is loaded, sequential re-writing of data from the storage means to the memory module may continue without interruption and without duplication of the requested segments.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a functional diagram illustrating a system for writing data from a storage means to memory in a solid state disk system, in accordance with the present invention.

FIG. 2 is a flow diagram illustrating steps of a method for writing data from a storage means to memory in a solid state disk system, which steps are implemented via the control module illustrated in FIG. 1.

FIG. 3 is a flow diagram illustrating steps of a method for writing data from a storage means to memory in a solid state disk system, which steps are implemented via the interface module illustrated in FIG. 1.

DETAILED DESCRIPTION

Referring now to the figures, the present invention is directed to a system and method for writing data from a storage means to a memory module in a solid state disk system, upon start-up. FIG. 1 is a block diagram illustrating the component parts of the invented system and the function of each component part. The invented system comprises a solid state disk system **101** having a non-volatile storage means **102**, a control module **103**, a memory portion **104**, an interface module **105** that communicates with external devices **106**, and an internal power supply **107**. The storage means **102** comprises a means for electronic storage of data that does not need to be periodically refreshed. The storage means **102** may comprise, for example, a hard disk system. The storage means may alternatively comprise a non-volatile storage means, such as a semiconductor memory array, a flash memory array, or a flash disk.

The control module **103** facilitates the copying of data to the storage means **102** from the memory module **104**, and the re-writing of data from the storage means **102** to the memory module **104**. The control module **103** automatically manipulates the copying and re-writing of data between the storage means **102** and the memory module **104**, according to the method described with reference to FIG. 2. The control module **103** also maintains a sequential listing of data segments, as stored by the storage means **102**, that denotes which segments have been written to memory. The control module also maintains a load priority queue for data segment requests that it receives from the interface module **105**. The control module **103** may also allow for manual manipulation of the copying and re-writing of data. The control module **103** preferably comprises a microprocessor-controlled solid state disk controller, the elements of which are readily known to those skilled in the art.

The memory portion **104** comprises at least one direct-access memory module for holding data in current or recent use by external devices.**106**. The memory module **104** is more quickly accessible, and performs read and write processing functions more quickly, than the non-volatile storage means **102**. The memory portion **104** preferably comprises at least one random-access memory (RAM) module. A RAM module may comprise dynamic random-access memory (DRAM), synchronous DRAM (SDRAM) or other appropriate technology.

The interface module **105** manages communication with external devices **106**. The interface module **105** processes data requests from external devices **106** and sends and receives commands with the control module **106**. With respect to the current invention, the interface module **105** receives requests from external devices **106** for particular data blocks. The communication module translates these blocks into segments, sends data requests to a load priority queue maintained by the control module **103** and receives data segment availability notices from the control module **103** after each data segment is written from storage means **102** to memory **104** after start-up.

The interface module **105** also processes read and write commands from memory **104**, in order to retrieve requested data segments from memory **104** and return them to external devices **106**. The interface module **105** may communicate with external devices **106** via Ethernet, SCSI or FC. Preferably, the interface module **105** communicates with external devices **106** via FC. The interface module **105** may comprise a general integrated circuit and associated circuitry, or may comprise several integrated circuits and associated circuitry, such that it may process requests from multiple applications. Alternatively, the interface module **105** may comprise an application-specific integrated circuit (ASIC), such as a QLogic Fiber Channel ASIC.

Alternatively, the interface module **105** and control module **103** may be combined on a single integrated circuit.

External devices **106** comprise computing devices having central processing units that are capable of submitting commands and data requests to, and receiving requested data from, the interface module **105** via FC communication, Ethernet, SCSI or other appropriate communication means.

The internal power supply **107** comprises a temporary power supply suitable for providing adequate power to facilitate the copying of data from the memory **104** to the storage means **102** in the event that external power to the system should fail. The internal power supply **107** may comprise, for example, at least one battery, extended-life battery pack or direct current uninterrupted power supply (DC UPS). Upon shutdown or failure of external power to the system **101**, the internal power supply **107** provides sufficient power for data residing in memory **104** to be copied to the storage means **102**, upon prompting by the control module **103**. When power is restored and start-up of the system **101** is initiated, the data is re-written from the storage means **102** to the memory **104** in accordance with the present invention.

FIG. 2 is a flow diagram illustrating steps of a method for writing data from a storage means to memory in a solid state disk system, which steps are implemented via the control module illustrated in FIG. 1. In accordance with step **201**, shutdown or failure of the system's external power source occurs. The system maintains power via its battery, as described above, and the control module copies data in memory to the storage means, in accordance with step **202**. The data is copied in segments that are stored sequentially on the storage means. Once external power is restored to the system at step **215**, the start-up process is initiated, in accordance with

step **203**. In accordance with step **204**, the control module creates a sequential load map, which lists the data segments stored on the storage means, in the order they are stored.

In accordance with step **205**, the control module then determines whether all segments in the sequential load map have been written to memory. When all segments have been written to memory, the start-up process terminates, in accordance with step **213**. Where segments exist in the sequential load map that have not been written to memory, the start-up process does not terminate and proceeds to step **206**. Those skilled in that art will appreciate that upon the initiation of the start-up process, all data segments in the sequential load map will not have been written to memory.

If the start-up process is not terminated, then the control module receives any requests from the interface module for particular data segments stored on the storage means, in accordance with step **206**. The manner in which the interface module sends such requests is described further with reference to FIG. 3, below. The control module places these requests in a load priority queue comprising a plurality of priority positions. Request may be placed in priority positions in the order they are received, such that the first received request is placed in the highest priority position.

In accordance with step **207**, the control module determines whether there are any requests for segments in the load priority queue. If at least one request is in queue, then the control module loads the data segment corresponding to the request in the highest occupied priority position of the queue, in accordance with step **208**. The requested data segment is then written to memory, in accordance with step **210**, and the segment is marked as active on the sequential load map, in accordance with step **211**. The interface module is then notified of the segment's availability in memory, in accordance with step **212**.

Where no requests exist in the load priority queue in step **207**, then the control module loads the next data segment in the sequential load map, in accordance with step **209**. This segment is then written to memory, in accordance with step **210**, and the segment is marked as active on the sequential load map, in accordance with step **211**. The interface module is then notified that the segment has been loaded into memory, in accordance with step **212**.

After the control module notifies the interface module that a data segment is available in memory, the control module then checks to see if all data segments in the sequential load map have been written to memory, in accordance with step **205**. This may be done either by comparison of data between the storage means and memory module(s), or by checking whether all segments listed on the sequential load map are marked as active. The method then repeats beginning with step **205** through step **212**, until all data segments have been loaded to memory and the start-up process terminates in accordance with step **213**.

FIG. 3 is a flow diagram illustrating steps of a method for writing data from a storage means to memory in a solid state disk system, which steps are implemented via the interface module illustrated at **105** in FIG. 1. After power is restored to the system in accordance with step **301**, as described above, the communications management module creates a log for recording which data segments are available in memory, in accordance with step **302**. Alternatively, the communications management module may simply re-initialize a pre-created log.

In accordance with step **304**, the interface module receives a request from an external device for at least one block of data. In accordance with step **305**, the interface module translates the requested blocks into their locations within data segments stored by the storage means or stored in memory. The interface module may perform this function by communicating with the

control module and searching for the requested blocks in the sequential load map described above. Alternatively, the interface module may abstract a copy of the sequential load map from the control module, in accordance with step **303**.

In accordance with step **306**, the interface module checks to see whether the segment(s) containing the requested block(s) of data are available in memory, as recorded in the interface module's segment availability log. If the segment(s) are available, then the interface module retrieves the segment(s) from memory, in accordance with step **309**. The interface module then extracts the requested data block(s) from the segment(s) and returns them to the external device, in accordance with step **310**.

If the interface module determines that the needed segments are not available in memory, in accordance with step **306**, then the interface module sends a request for the segments to the load priority queue maintained by the control module, in accordance with step **307**. The interface module may issue separate requests for each block contained in a request received from the external device. Alternatively, the interface module may submit a single request for all blocks requested by the external device.

Once all segments contained in a request sent by the interface module have been copied into memory by the control module, according to the steps described with reference to FIG. 2, the interface module receives a notice from the control module, in accordance with step **308**. The notice informs the interface module that the requested segment(s) have been copied to memory, and their availability is recorded by the interface module in its segment availability log. The interface module then retrieves the segment(s) from memory, in accordance with step **309**. The interface module then extracts the requested data block(s) from the segment(s) and returns them to the external device, in accordance with step **310**.